**Tema 3**

**Order Management**

Mesaros Radu-Andrei

30223

**Cuprins**

1. Obiectivul temei: pagina 3

2. Analiza Problemei: pagina 3

3. Proiectare: pagina 4

4. Implementare: pagina 5 – pagina 8

5. Rezultate: pagina 9

6. Concluzii: pagina 9

7. Bibliografie: pagina 10

**Obiectivul temei**

Proiectarea unui program care se ocupa cu gestionarea si procesarea unor comenzi plasate de niste clienti, intr-un anumit magazin.

Obiectivul principal este de a introduce, sterge, modifica si afisa date din baza de date a magazinului

Ca obiective secundare pentru a îndeplini obiectivul principal avem:

* + - Transformarea datelor de intrare introduse de către utilizator în date de program care pot fi folosite în efectuarea operațiilor..
    - Modelarea datelor de intrare astfel incat sa putem opera cu baza de date si sa efectuam operatiile necesare
    - Realizarea conexiunii programului cu baza de date.
    - Crearea unei chitante pentru fiecare comanda noua.

**Analiza problemei**

Utilizatorul foloseste o interfata grafica pentru a introduce datele pentru operatiile de inserare, modificare sau stergere de date din baza de date iar apoi prin apasarea butoanelor corespunzatoare se efectueaza operatiile de inserare, stergere, modificare si afisare a datelor.

**Proiectare**

Am folosit 3 clase Model: Client, Product si Order. Acestea modeleaza conceptul de client, produs si comanda. Pentru conexiunea cu baza de date am folosit clasa ConnectionFactory. Pentru interactiunea cu baza de date am folosit clasele ClientDAO, ProductDAO, OrderDAO, toate mostenind clasa AbstractDAO. Pentru verificari si alte functionalitati am folosit clasele ClientBL, ProductBL si OrderBL. Pentru validari de date am folosit clasa EmailValidator si PhoneNumberValidator care implementeaza interfata Validator. Clasa Controller gestioneaza modul in care functioneaza programul, iar clasa View are interfata grafica si se ocupa de comunicarea cu utilizatorul. In clasa Main se realizeaza pornirea programului.

A computer screen shot of a computer program

Description automatically generated

Pentru implementarea acestui program am folosit array-uri simple de una si doua dimensiuni si liste simple de obiecte. Ca tehnici de implementare am folosit tehnica de reflexion pentru a crea o clasa care contine metodele generice pentru lucrul cu baza de date.

**Implementare**

**Clase**

* **View** – aceasta clasa se ocupa cu interfata grafica, contine text label-uri, text field-uri si mai multe butoane. De asemenea contine cateva tab-uri, pentru inserare, editare si stergere de clienti sau produse. Mai are si doua combo-box-uri pentru a putea selecta un client si un produs, ca sa se efectueze comanda.
* **Controller** – se ocupa de functionarea programului. Contine clasele care implementeaza ActionListener-uri pentru butoanele de pe interfata
* **Client** – clasa aceasta reprezinta un Client, o persoana din viata reala, are un id unic, un nume, o adresa, un numar de telefon si un email. Email-ul si numarul de telefon sunt verificate astfel incat sa nu fie invalide.
* **Product** – reprezinta un produs din depozit, are un id unic, un nume, un producator, un pret si o cantitate.
* **Order** – reprezinta o comanda de la magazin, are id unic, are id-ul unui client si id-ul unui produs.
* **Email/PhoneNumber Validator** – in aceste 2 clase se realizeaza validarea datelor
* **ConnectionFactory** – este clasa care se ocupa de conexiunea cu baza de date
* **ClientBL** – apeleaza metodele de inserare, editare, stergere si afisare din ClientDAO. Aici mai au loc si alte verificari si validari de date.
* **ProductBL** - Clasa ProductBL este clasa care se ocupa de apelarea metodelor de inserare, modificare, stergere si afisare din clasa ProductDAO.
* **OrderBL** - Clasa OrderBL este clasa care se ocupa de apelarea metodelor de inserare si afisare a datelor din tabelul Orders din baza de date folosind clasa OrderDAO.

**Interfata Grafica**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Aici este tab-ul pentru inserarea unui Client, se introduc datele si se apasa Add new Client. Daca apasam Delete Client, va sterge clientul cu id-ul pe care l-am scris noi in text box, de asemenea daca apasam pe Display Clients, se vor afisa toti clientii cu datele acestora.

Tab-ul de la Product arata si functioneaza analog acestui tab.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Daca se introduce un client cu un id care deja exista, se va afisa mesajul de eroare. La fel, daca se introduce un email/nr de telefon gresit, se va afisa un mesaj de eroare.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedLa tab-ul de Order avem cele 2 text box uri, selectam clientul, produsul si scriem cantitatea dorita. Dupa care se va executa comanda, se va scadea din stock ul existent si se va printa o factura. De asemenea, daca nu sunt suficiente produse in stoc se va afisa un mesaj de eroare.

**Rezultate**

A screenshot of a computer

Description automatically generated

Aici avem o chitanta produsa la crearea unei comenzi unde se poate vedea data si ora la care s-a efectuat comanda, numele clientului, produsul, pretul pentru o bucata, cate produse a cumparat clientul si costul total al comenzii.

**Concluzii**

Dezvoltarea acestei teme m-a ajutat sa invat cum se lucreaza cu baze de date. Mai exact cum se stabileste o conexiune cu o baza de date createa in MySQL Workbench de exemplu, cum se pot implementa metodele de inserare modificare sterge si afisare a datelor din baza de date.

Idei de imbunatatire a programului sunt numeroase deci voi enumera doar cateva dintre ele:

• Posibilitatea de a introduce doar campurile ce trebuie modificate nu reintroducerea tuturor campurilor pentru operatiile de modificare a datelor in tabel

• Crearea mai multor clase de validare si protejarea executiei programului intr-un mod mai puternic, tratarea mai multor cazuri speciale si returnarea unor mesaje de eroare mai explicite

• O interfata grafica mai stilizata sim ai ordonata, mai ales in tab-ul pentru crearea de comenzi

**Bibliografie**

<http://tutorials.jenkov.com/java-reflection/index.html>

<https://www.baeldung.com/javadoc>

<https://gitlab.com/utcn_dsrl/pt-layered-architecture>

<https://gitlab.com/utcn_dsrl/pt-reflection-example>

<https://chatgpt.com/>

<https://dsrl.eu/courses/pt/>